AMENDMENTS TO THE CLAIMS

Please amend claims 28 and 49 and cancel claims 26-27 and 35, such that the status of the claims is as follows:

1-27. (Canceled)

28. (Currently Amended) An architecture for developing a distributed information system, the architecture comprising:

a component development tool for generating a plurality of autonomous and compiled components that implement and consume services, the components capable of operating in an edit mode and a run mode;

a system development tool for defining and hosting a plurality of component instances based on the plurality of components, configuring the plurality of component instances, and defining links between component instances, without requiring writing of code, wherein the component instances are capable of operating in the edit mode while hosted by the system development tool; and

an engine software program to provide a dynamic run-time environment for hosting the plurality of component instances and supporting communication between component instances based upon the defined links, wherein the component instances are capable of operating in the run mode while hosted by the engine software program.

29. (Previously Presented) The architecture of claim 28, further comprising:

a service definition tool for generating service protocols that are implemented by components, the service protocols defining a format of messages to be sent between ports of component instances; and

wherein links defined between component instances are defined between the ports of component instances.

30. (Previously Presented) The architecture of claim 28, wherein the component development tool provides the capability of representing components as a first and a second plurality of components, each component in the first plurality of components representing a physical entity in the distributed information system, and each component in the second plurality of components representing a logical entity in the distributed information system.

31. (Previously Presented) The architecture of claim 29, wherein each of the ports comprises either a service provider port or a service consumer port.

32. (Previously Presented) The architecture of claim 31, wherein service provider ports and service consumer ports based on the same service protocol are complementary.

33. (Previously Presented) The architecture of claim 32, wherein the system development tool only allows links to be defined between service provider ports and complementary service consumer ports.

34. (Previously Presented) The architecture of claim 28, wherein each of the plurality of component instances is self-sufficient.

35. (Canceled)

36. (Previously Presented) The architecture of claim 33, wherein a component instance includes at least one service provider port that allows multiple simultaneous links with complementary service consumer ports.

37. (Previously Presented) The architecture of claim 33, wherein component instances are executed concurrently, and wherein the communications between service provider ports and complementary service consumer ports are asynchronous.

38. (Previously Presented) The architecture of claim 28, wherein the engine software program runs on each of a plurality of networked nodes.

39. (Previously Presented) The architecture of claim 38, wherein the system development tool represents the distributed information system as a single entity, regardless of physical node and network composition into which the component instances will be deployed.

40. (Previously Presented) The architecture of claim 38, wherein the system development tool deploys each component instance to one of the plurality of networked nodes.

41. (Previously Presented) The architecture of claim 38, and further comprising a local repository on each of the plurality of nodes, the local repository on each node storing data defining the component instances deployed to and hosted by that node and storing link data for the component instances deployed to and hosted by that node.

42. (Previously Presented) The architecture of claim 39, wherein the system development tool allows changes to be made to the component instances deployed to and hosted by the plurality of networked

nodes and allows changes to be made to links between the component instances deployed to and hosted by the plurality of networked nodes.

43. (Previously Presented) The architecture of claim 42, wherein the system development tool allows changes to be made to the component instances and allows changes to be made to links between the component instances, without requiring writing of additional code, wherein the system development tool allows the changes to be made while the distributed information system is running.

44. (Previously Presented) The architecture of claim 43, wherein the system development tool allows deletion of the component instances deployed to and hosted by the plurality of networked nodes and allows deletion of communication links between the component instances deployed to and hosted by the plurality of networked nodes, wherein the system development tool allows the deletions to occur while the distributed information system is running.

45. (Previously Presented) The architecture of claim 29, and further comprising a central system repository for storing the components, the component instances, link data, infrastructure configuration and configuration data for the service protocols.

46. (Previously Presented) The architecture of claim 28, wherein at least one of the component instances supports continuous activities internally.

47. (Previously Presented) The architecture of claim 28, wherein each of the component instances is configurable to participate in activities that are collectively performed by multiple component instances.

48. (Previously Presented) The architecture of claim 28, wherein the only dependencies between component instances that are linked to each other are logical dependencies implemented using the component development tool.

49. (Currently Amended) An architecture for developing a distributed information system, the architecture comprising:

a component development tool for generating autonomous and compiled components that implement and consume services, the components capable of operating in an edit mode and a run mode;

a system development tool for defining and hosting a plurality of component instances based on the plurality of components, configuring the plurality of component instances, and defining links between component instances, wherein the component instances are capable of operating in the edit mode while hosted by the system development tool; and

an engine software program to provide a programmable run-time environment for hosting the plurality of component instances and implementing the links to provide bi-directional communication paths between the plurality of component instances, wherein the component instances are capable of operating in the run mode while hosted by the engine software program.

50. (Previously Presented) The architecture of claim 49, further comprising a service definition tool for generating service protocols which define a format of messages to be sent through a plurality of ports, each port being associated with a component instance.

51. (Previously Presented) The architecture of claim 49, wherein the engine software program provides the bi-directional communication paths between linked ports.

52. (Previously Presented) The architecture of claim 49, further comprising a plurality of networked nodes running the engine software program, wherein the engine software dynamically manages ports and links for the component instances across the plurality of networked nodes.

53. (Previously Presented) The architecture of claim 49, wherein the component development tool is designed to be operated by a person skilled in computer programming.

54. (Previously Presented) The architecture of claim 49, wherein the system development tool is designed to be operated by a person without skill in the art of computer programming.

55. (Previously Presented) The architecture of claim 49, wherein any component instance having a consumer port that complies with a first service protocol may be configured to communicate with any component instance having a provider port that also complies with the first service protocol.